Time of Arrival via Onset of Non-Noise Signal Frequencies

Thomas J. Asaki
MS D413
Los Alamos National Laboratory
Los Alamos, NM, 87545
email: asaki@lanl.gov
phone: 505-667-5787

Other contributors include Bill Allard (Duke) and Robert Owczarek (LANL). Additional help was obtained through DDMA team discussions.

Description

The goal of this algorithm is to detect the time of arrival of a signal of interest in a 1D noisy background time series. The basic concept is to monitor non-noise frequency components using a windowed FFT scheme. The time of arrival of the signal should correspond to the time of arrival of non-noise frequencies.

One potential benefit of working in the frequency space is that the signal to noise ratio can be greater. A second benefit is that one can monitor any subset of frequencies that are considered relevant. Potential difficulties include the need for significant pre-signal data, the presence of non-noise features in the pre-signal, and the use of a user-determined sensitivity parameter.

Mathematical Principles

Consider the sequence $A = \{a_1, a_2, ..., a_n\}$ of real-valued signal amplitudes and the corresponding sequence of time stamps $T = \{t_1, t_2, ..., t_n\}$, where $t_i < t_{i+1}$. For our discussion we will use the unitless time convention $t_i = i$. We define subsequences $A_{i,j} = \{a_i, a_{i+1}, ..., a_j\}$ of length $h = j - i + 1$. For example, $A_{1,n} = A$. We also define the element notation on sequences $A_{i,j}$ as $A_{i,j}(k) = a_{i+k-1}$. An synthetic example time sequence is shown in Figure **??**. In this example, $n = 4500$ and the signal onset time is at t_{165}. Zero mean Gaussian noise of variance 2 corrupts an otherwise clean signal of fourier components of frequencies $\omega_g = 2\pi(3g - 1), k = 1, ..., 10$ and an envelope function $(t - t_{165})exp(-(t - t_{165})/120)$.

We next consider the FFT signature on sequences $A_{i,j}$. Let $B_{i,j} = mod(FFT(A_{i,j})) = \{b_1, b_2, ..., b_h\}$. The b_k are the moduli of the components returned by a fast fourier transform of $A_{i,j}$ with corresponding frequencies $\omega_k = 2\pi(k - 1)/h$. Thus, b_1 is the zero frequency amplitude offset of $A_{i,j}$, b_2 the lowest frequency signature given the sequence size h, etc. The initial portion of the b_2 signature of the example time sequence is shown in Figure **??**(top) for $h = 64$. The signal onset is clearly shown as a rise in b_2 near the expected onset time t_{165}.

To arrive at an objective procedure for determining the time of arrival, t_0, we define the contribution sequence $C_{i,j}$ with elements $C_{i,j}(k) = |(B_{i,j}(k) - \bar{B}_{i,j-1}(k))|/\bar{B}_{i,j-1}(k)$, where $\bar{B}_{i,j-1}(k) = (1/(j - h + 1)) \sum_{p=h}^{j} B_{p-h+1,p}$. The contribution sequence $C_{i,j}$ is the relative deviation in $B_{i,j}$ from the running mean of $B_{i,j}$. A portion of $C_{i,j}$ for the example sequence at $k = 2$ and $h = 50$ is shown in Figure **??**(bottom). Time of arrival extraction requires two steps. First, we compute the sequence $C_{i,j}(k)$ for $k = h - 1, ..., k*$ until $C_{i,j}(k)$ exceeds a sensitivity threshold. Figure **??**(bottom) shows the results of these calcuations with a sensitivity threshold value of 3. Second. we use the longest subsequence $\{b_j, ...b_{k*}\}$ such that $b_q < b_{q+1}, j \leq q < k*$ to provide a linear extrapolation to the time axis. The time intercept of this line is the time of arrival. We round the result to the nearest sequence time so that $t_0 \in T$. This linear extrapolation for the example is shown as a red line in Figure **??**(top). In this case the actual time of arrival was recovered $t_0 = 165$.

A quick test was performed on strain gauge data from threaded assembly experiments. The time of arrival results for twelve scenarios are shown in Figure /reffig:toa. Blue circles show the toa computed by this algorithm from raw *.wft data with nondefault settings of toff=100000 and sens=6. The red circles show the expert oppinion times of arrival.

The above algorithm can be modified to account for known data features and expert knowledge. Some discussion is given in the section on usage.

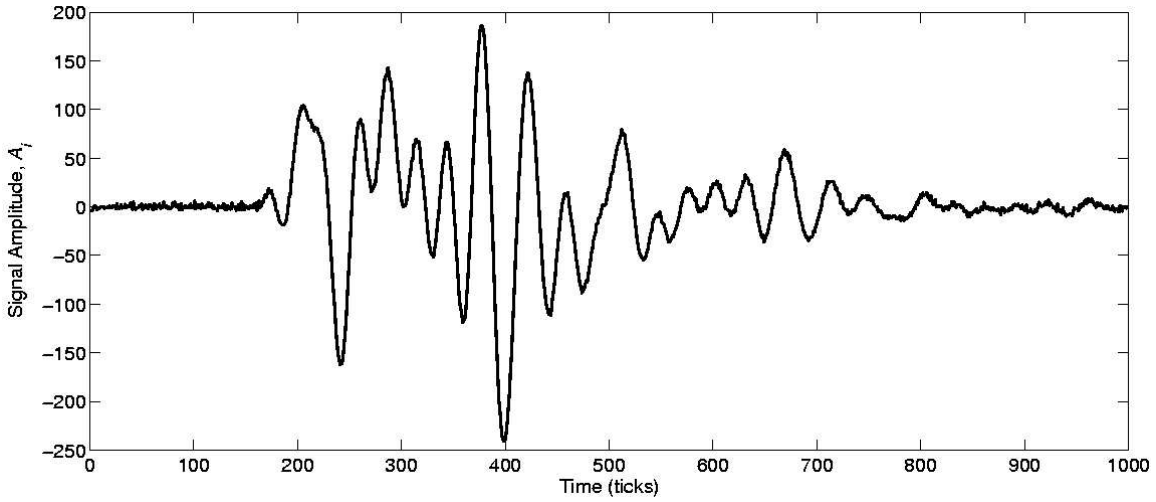

Figure 1: Example synthetic time trace with Gaussian noise. The time of signal onset is $t_0 = 165$.

Physical and Engineering Principles

This time of arrival algorithm relies on a few important assumptions, none of which are unique to the Threaded Assembly data.

First, the signal of interest is assumed to begin at a time when the frequency content of the time series just begins to contain non-noise-like characteristics. This frequency signature can be known or assumed. For accelerometer and strain gauge data initially in a quiescent state, this assumption is certainly valid.

Second, the time series should be prepended with data that does not contain frequencies of interest in time of arrival detection. Offsets, linear drift, and frequencies not near ω_k should not significantly affect algorithm performance.

For accelerometer and strain gauge data initially in a quiescent state, these assumptions are certainly reasonable.

Usage

The Matlab script that runs the algorithm is reproduced below. The comments and notations address parameter selection. It is important to note that default parameter values should work very well for many scenarios.

```
function [toa]=TOA(data,sens,toff,wind,freq,lsfg);

%
% TOA returns the time of arrival of a signal within a
% data set.  Tom Asaki (667-5787) (asaki@lanl.gov)
%
% EXAMPLE USAGE:
% [toa]=TOA(data)
% [toa]=TOA(data,sens)
% [toa]=TOA(data,[],toff)
% [toa]=TOA(data,sens,toff,wind,freq,lsfg)
%
% INPUTS:
% data    vector of data values treated as as amplitudes
%         in a time sequence.
% sens    a sensitivity factor for determining when low-
%         frequency components are present above noise
%         level in the frequency signature.  Default = 3.
```

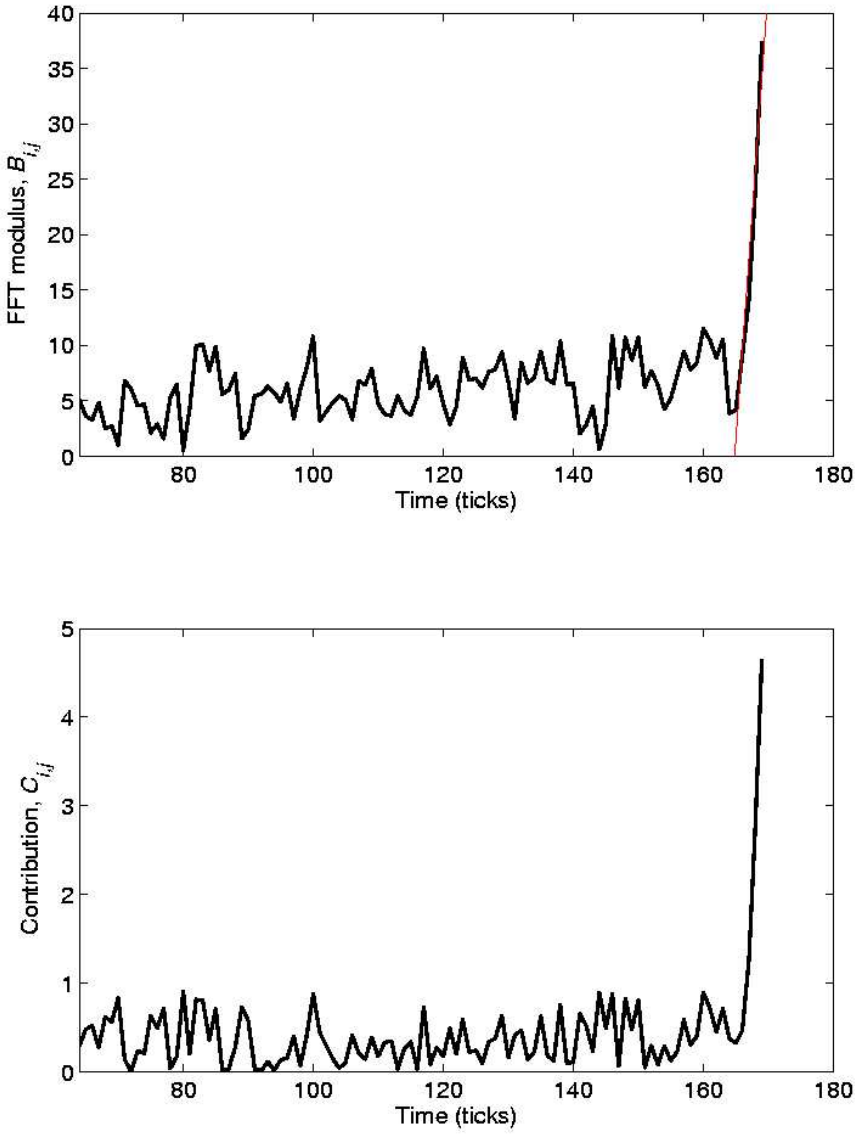

Figure 2: The top figure shows the running windowed FFT moudlus for the lowest frequency using a window size of 64. The bottom figure is the corresponding contribution sequence defined in the text. The red line shows the extrapolation to the time axis for determining the time of arrival.

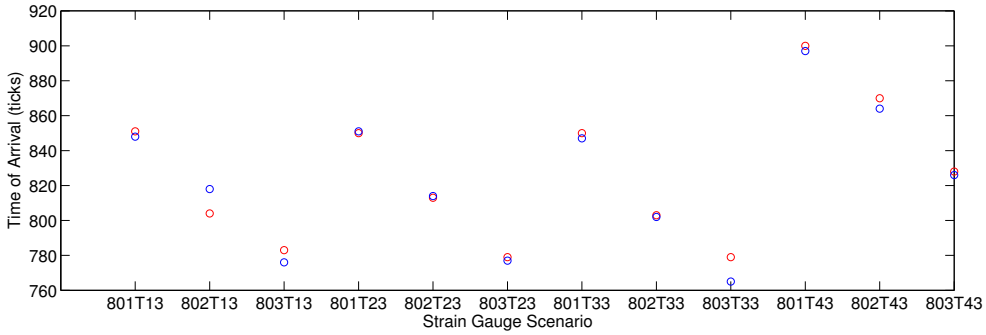

Figure 3: Time of arrival results for threaded assembly strain gauge data. Blue = calculated, red = expert oppinion.

```
%         Typical ranges are 2 to 4 and sens must be >=1.
%         sens=[] is equivalent to setting the default.
%         sens operates counterintuitively in that lower
%         values lead to signal onset determinations that
%         are more sensitive to noise fluctuations,
%         leading to earlier toa.
% toff    nonnegative time step offset.  This can be used
%         to ignore early values for either reasons of
%         compuational time or the presence of early time
%         signal features the user wishes to ignore.
%         Default = 0.  toff=[] is equivalent to setting
%         the default.
% wind    window size over which the frequency content of
%         the signal is examined.  The user can select
%         this size to reduce the effects of noise.  The
%         user can also modify the size to select a
%         particular low frequency of interest (along with
%         freq variable described below).  wind >= 8.
%         Default = 64.  wind=[] is equivalent to setting
%         the default.
% freq    integer for selecting the frequency component of
%         interest from the fft.  freq = 1 corresponds to
%         the zero frequency amplitude offset, freq=2
%         corresponds to the low frequecy determined by
%         the size of wind, etc.  1 <= freq <= wind.
%         Default = 2.  freq=[] is equivalent to setting
%         the default.
% lsfg    linear suppression flag.  lsfg set to ~0 means
%         that the algorithm will subtract the linear
%         contribution from the windowed data before
%         computing the fft.  Default=1.  lsfg = [] is
%         equivalent to setting the default.
%
% OUTPUTS:
% toa     the time of arrival of the signal in units of
%         signal ticks -- the signal is deemed to arrive
%         at the toa'th value in the input signal.  If no
%         time of arrival was reliably determined then
%         toa is returned as 0.
%
% NOTES:
% (1) It seems reasonable to automate a default choice for
% both freq and wind based on the fft of the entire signal.
% This is not complete and maybe not useful.
% (2) A default sens is more difficult but might be
% extracted from a windowed fft signal taken over a large
% domain.
% (3) This method is sensitive to the amount of presignal
% (before onset).  That is, it requires that the presignal
% be larger than wind by a factor of at least 2 for
% reliable results.
% (4) This method will be fooled by presignal features.
% toff should be used to take care of this, or data should
% be pre-truncated or preprocessed.
% (5) lsfg should be used with caution.  If the pre-signal
% is expected to be corrupted by linear or very low
```

```
% frequency drift then lsfg=1 may be helpful.
%
% ALGORITHM:
% (1) compute windowed ffts on the data at one tick
% increments.
% (2) concurrently compute the running mean of this fft
% value
% (3) continue the first two steps until the sensitivity
% requirement is met: freq amplitude is sens times the
% mean above the mean.  This is the criteria that onset
% is happening.
% (4) use recent fft values to back out toa by linear
% extrapolation.  The points use are those corresponding
% to increasing values of the test in step 3.
%

%%%%% INITIALIZATIONS %%%%%%%%%%%%%%%%%%%%%%%%%

%%% define default values on inputs
sens_def=3;
toff_def=0;
wind_def=64;
freq_def=2;
lsfg_def=1;

%%% set default values of inputs
if nargin<6 ; lsfg=lsfg_def ; end
if nargin<5 ; freq=freq_def ; end
if nargin<4 ; wind=wind_def ; end
if nargin<3 ; toff=toff_def ; end
if nargin<2 ; sens=sens_def ; end
if length(lsfg)==0 ; lsfg=lsfg_def ; end
if length(freq)==0 ; freq=freq_def ; end
if length(wind)==0 ; wind=wind_def ; end
if length(toff)==0 ; toff=toff_def ; end
if length(sens)==0 ; sens=sens_def ; end

%%% fix input parameter values
freq = round(min(wind,max(2,freq)));
wind = round(max(8,wind));
toff = round(max(0,toff));
sens = max(1,sens);
lsfg=lsfg~=0;

%%% other stuff
shft=toff+wind-1;
data=data(:);

%%%%% MAIN ROUTINE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% search for onset
k=0;
ftest=1;
etest=1;
while ftest & etest
    k=k+1;
```

```
    if lsfg
        [x,y,m,b]=fitline(0:wind-1,data(k+toff:k+shft));
        fftdata(:,k)=abs(fft(data(k+toff:k+shft)-y));
    else
        fftdata(:,k)=abs(fft(data(k+toff:k+shft)));
    end
    fftdatamean(k)=mean(fftdata(freq,1:k));
    onsetfactor(k)=abs(fftdata(freq,k)-fftdatamean(k))/fftdatamean(k);
    ftest=onsetfactor(k) < sens;
    etest=(k+shft) < length(data);
end
diffonsetfactor=diff(onsetfactor);

%%% modify onset by back extrapolation
if etest
    kk=k;
    while diffonsetfactor(kk-1)>0
        kk=kk-1;
    end
    [xx,yy,m,b] = fitline(kk:k,fftdata(freq,kk:k));
    timeintercept=-b/m;
    toa=round(timeintercept+shft);
else
    toa=0;
end

return

%%%%% INTERNAL FUNCTIONS %%%%%%%%%%%%%%%%%%%%%%%%

function [xx,yy,m,b] = fitline(x,y)
% Subroutine FITLINE computes the least squares linear
% fit to the coordinates (x,y).  It returns the line
% slope (m) and intercept (b) such that y=mx+b, and
% discete line description (xx,yy) where xx=x(:).
A=[ones(size(x(:))) x(:)];
c=A\y(:);
b=c(1);
m=c(2);
xx=x(:);
yy=b+m*x(:);
return

%%%%% END %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```